bio-protocol Preprint

Not Peer Reviewed
Ask a question online

Single-cell bioinformatics analysis for thyroid cancer study

Han Luo Gyeong Dae Kim Tao Wei Jihwan Park Heng Xu

Updated date: Aug 6, 2021

An abbreviated version of this protocol was published in Science Advances in Jul 2021

Characterizing dedifferentiation of thyroid cancer by integrated analysis

DOI: 10.1126/sciadv.abf3657

Detailed protocol

```
### Single-cell bioinformatics analysis for thyroid cancer study

#-------------------------------------------------------------------------------
### Integration of scRNAs-seq data
#-------------------------------------------------------------------------------

# dependency package install

devtools::install_github('satijalab/seurat-wrappers',force = T)
need_pkgs <- c('Seurat','cowplot', 'slingshot',
               'data.table', 'reticulate','dplyr',
              'RColorBrewer','SingleCellExperiment')


for(i in need_pkgs){
  require(i, quietly = T, character.only = T)
}


# setting the file pathway

setwd('/home/gyeongdaekim/')
file_list <- list.files()
print(file_list)

sampleNames = mapply(function(x) x[length(x)], strsplit(file_list, split = '_'))
sampleNames <- gsub('-','_',sampleNames)
meta.info = data.table(file_Path = file_list, sampleID = sampleNames)

meta.info[, SubType:=mapply(function(x) x[1], strsplit(sampleID, '_'))]
DT::datatable(meta.info)


# setting the integration and QC option

norm_method = 'standard'
treat = FALSE
show = FALSE
MT_cut = 15
batch_list = list()

for(i in 1:nrow(meta.info)){
  dir_of_10X = meta.info$file_Path[i]
  sampleID = meta.info$sampleID[i]
  subtype = meta.info$SubType[i]
  seq_batch = meta.info$Batch[i]
  print(paste("Starting processing", sampleID, "at", Sys.time()))
```

```
  print(paste("Starting processing", sampleID, "at", Sys.time()))
  sc_mat = Read10X(dir_of_10X)

  sc_tumor <- CreateSeuratObject(counts = sc_mat, project = sampleID,
                                 min.cells = 3, min.features = 200)
  sc_tumor[["percent.mt"]] <- PercentageFeatureSet(sc_tumor, pattern = "^MT-")

  qc = VlnPlot(sc_tumor, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)

  plot1 <- FeatureScatter(sc_tumor, feature1 = "nCount_RNA", feature2 = "percent.mt")
  plot2 <- FeatureScatter(sc_tumor, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")

  sc_tumor <- subset(sc_tumor, subset = nFeature_RNA > 200  & percent.mt <= MT_cut)

  if(norm_method == 'SCT'){
    sc_tumor <- SCTransform(sc_tumor, vars.to.regress = 'percent.mt')
  } else{
    sc_tumor <- NormalizeData(sc_tumor, verbose = F)
    sc_tumor <- FindVariableFeatures(sc_tumor, selection.method = "vst",
                                     nfeatures = 2000, verbose = FALSE)

    sc_tumor <- ScaleData(sc_tumoer, vrbose = F)
  }

  sc_tumor <- RenameCells(object = sc_tumor, add.cell.id = sampleID)  # add sample name as prefix
  if(treat == TRUE){
    sc_tumor <- RunPCA(sc_tumor, verbose = T)
    sc_tumor <- RunUMAP(sc_tumor, dims = 1:30, verbose = FALSE)
    sc_tumor <- RunTSNE(sc_tumor, dims = 1:30, verbose = FALSE)

    sc_tumor <- FindNeighbors(sc_tumor, dims = 1:30, verbose = FALSE)
    sc_tumor <- FindClusters(sc_tumor, verbose = FALSE)

    umap = DimPlot(sc_tumor, label = TRUE) + NoLegend()
    tsne = TSNEPlot(sc_tumor, label = TRUE) + NoLegend()

    if(show == TRUE){

      print(qc)
      plot_grid(plot1,plot2)
      #dev.off()

      print(umap)
      print(tsne)
    }
  }

  # add information
  sc_tumor@meta.data[,'SubType'] = subtype
  sc_tumor@meta.data[,'SampleID'] = sampleID
  sc_tumor@meta.data[,'Batch'] = seq_batch
  ### add batch data into a list to merge
  #assign(paste0(sampleID,'.Seurat'),sc_tumor)
  batch_list <- append(batch_list, sc_tumor)
  print(paste("Finishing processing", sampleID, 'at', Sys.time()))
}

data.combined <- Reduce(function(x,y){merge(x,y,merge.data = TRUE)},batch_list)


# QC check
qc = VlnPlot(data.combined, group.by = 'SampleID', features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3,pt.size = 0)
qc



#-------------------------------------------------------------------------------------------
### scRNA-seq data analysis using the Seurat
#-------------------------------------------------------------------------------------------


library(Seurat)
```

```
library(Seurat)

data.combined <- FindVariableFeatures(data.combined, selection.method = "vst", nfeatures = 2000, verbose = FALSE)

data.combined <- ScaleData(data.combined, vars.to.regress = c('nCount_RNA', 'percent.mt'),verbose = F)

data.combined <- RunPCA(data.combined, npcs=30, verbose=F)


# selection of the principle component number

data.combined <- JackStraw(data.combined, num.replicate = 100)

data.combined <- ScoreJackStraw(data.combined, dims = 1:20)

JackStrawPlot(data.combined, dims = 1:20)

ElbowPlot(object = data.combined)


num_PC = 30

data.combined <- RunUMAP(data.combined, reduction = 'pca', dims=1:num_PC)

data.combined <- FindNeighbors(data.combined, reduction = 'pca', dims=1:num_PC)

data.combined <- FindClusters(data.combined, resolution = 1)


DimPlot(data.combined, label = TRUE) + NoLegend()
DimPlot(data.combined, label = TRUE,split.by = "orig.ident",ncol = 4) + NoLegend()



#-------------------------------------------------------------------------------------------
### batch correction using the fastMNN
#-------------------------------------------------------------------------------------------


library(SeuratWrappers)
library(SeuratData)

data.combined5 <- RunFastMNN(object.list = SplitObject(data.combined, split.by = "SampleID"))

data.combined5 <- RunUMAP(data.combined5, reduction = "mnn", dims = 1:30)

data.combined5 <- FindNeighbors(data.combined5, reduction = "mnn", dims = 1:30)

data.combined5 <- FindClusters(data.combined5, resolution = 0.7)


DimPlot(data.combined5, label = TRUE) + NoLegend()




#-------------------------------------------------------------------------------------------
### calculation of the number of differentially expressed genes(DEGs) between clusters and merge
#-------------------------------------------------------------------------------------------


for(j in 0:length(names(data.combined5@active.ident))){

 n=table(data.combined5@active.ident)

 ident=j

 n[names(n) %in% c(0:ident)]=0

 n=n[-1]

 n=n[n>0]
```

```
 o=i[i>0]
 ident2=names(o[1])
 bim <- FindMarkers(data.combined5, ident.1 = ident,  ident.2 =ident2, only.pos = F, test.use = "bimod")
 deg=bim[bim$p_val_adj < 0.01,]
 deg1=deg[abs(deg$avg_logFC) >= 1,]
 degs=dim(deg)
 degs1=dim(deg1)

 for(i in c(names(o[-1]))){
  bim <- FindMarkers(data.combined5, ident.1 = ident,  ident.2 = i, only.pos = F, test.use = "bimod")
  deg=bim[bim$p_val_adj < 0.01,]
  deg1=deg[abs(deg$avg_logFC) >= 1,]
  degs=cbind(degs,dim(deg))
  degs1=cbind(degs1,dim(deg1))
 }
 print(degs1)
}
```

 # merge the similar clusters having the difference of DEGs less than 10

```
data.combined5 <- RenameIdents(object = data.combined5,"0"="0",  "1"="0",  "2"="2",  "3"="0",  "4"="4",  "5"="5",  "6"="0",  "7"="7",  "8"="8",
"9"="9",  "10"="10",  "11"="11",  "12"="12",  "13"="13",  "14"="14",  "15"="15",  "16"="16",  "17"="17")
```

 # again, calculation of the number of DEGs between clusters

```
data.combined5 <- RenameIdents(object = data.combined5,"0"="0",  "2"="1",  "4"="2",  "5"="3",  "7"="4",  "8"="5",  "9"="6",
"10"="7", "11"="8", "12"="9", "13"="10",  "14"="11",  "15"="12",  "16"="13",  "17"="14",  "18"="15")
```

 # marker of each cluster

```
data.combined5.marker <- FindAllMarkers(data.combined5, max.cells.per.ident = 100, min.diff.pct = 0.3, only.pos = TRUE)
write.csv(data.combined5.marker,file = "marker.csv")

FeaturePlot(data.combined5, features = c("SLC34A2"), cols = c("yellow2","DARKGREEN","BLUE4"))
```

 # annotation each cluster

```
new.cluster.ids <- c("PTC", "NK cell",  "Macrophage",  "Follicular",  "T cell",  "Endothelium",  "Fibroblast1",  "Macrophage2",  "PTC2",
"Fibroblast2",  "B cell",  "Mast",  "ATC",  "Activated T cell",  "TAMC", "Parafollicular")

names(new.cluster.ids) <- levels(data.combined5)
data.combined5 <- RenameIdents(data.combined5, new.cluster.ids)
```

 # sort cell types

```
ident <- c("Follicular",  "PTC",  "PTC2",  "ATC",  "Parafollicular",  "Fibroblast1",  "Fibroblast2",  "Endothelium",  "NK cell",  "T cell",  "B
cell",  "Mast",  "Activated T cell", "Macrophage",  "Macrophage2",  "TAMC")

data.combined5@active.ident<- factor(x = data.combined5@active.ident, levels = ident)

DimPlot(data.combined5, label = TRUE ) + NoLegend()
```

```
DimPlot(data.combined5, label = TRUE ) + NoLegend()
DimPlot(data.combined5, label = TRUE, split.by = "SubType" ,  ncol = 4, label.size = 0) + NoLegend()



#-----------------------------------------------------------------------------------------------
### stacked vlnplot
#-----------------------------------------------------------------------------------------------


modify_vlnplot<- function(obj,
                          feature,
                          pt.size = 0,
                          plot.margin = unit(c(-0.75, 0, -0.75, 0), "cm"),
                          ...) {
  p<- VlnPlot(obj, features = feature, pt.size = pt.size, ... )  +
    xlab("") + ylab(feature) + ggtitle("") +
    theme(legend.position = "none",
          axis.text.x = element_blank(),
          axis.ticks.x = element_blank(),
          axis.title.y = element_text(size = rel(1), angle = 0),
          axis.text.y = element_text(size = rel(1)),
          plot.margin = plot.margin )
  return(p)
}

## extract the max value of the y axis
extract_max<- function(p){
  ymax<- max(ggplot_build(p)$layout$panel_scales_y[[1]]$range$range)
  return(ceiling(ymax))
}


## main function
StackedVlnPlot<- function(obj, features,
                          pt.size = 0,
                          plot.margin = unit(c(-0.75, 0, -0.75, 0), "cm"),
                          ...) {

  plot_list<- purrr::map(features, function(x) modify_vlnplot(obj = obj,feature = x, ...))

  # Add back x-axis title to bottom plot. patchwork is going to support this?
  plot_list[[length(plot_list)]]<- plot_list[[length(plot_list)]] +
    theme(axis.text.x=element_text(), axis.ticks.x = element_line())

  # change the y-axis tick to only max value
  ymaxs<- purrr::map_dbl(plot_list, extract_max)
  plot_list<- purrr::map2(plot_list, ymaxs, function(x,y) x +
                            scale_y_continuous(breaks = c(y)) +
                            expand_limits(y = y))

  p<- patchwork::wrap_plots(plotlist = plot_list, ncol = 1)
  return(p)
}

features <- cluster marker
StackedVlnPlot(obj = data.combined5, features = features)

save(data.combined5, file="/home/gyeongdaekim/thyroid/object/data.combined.RData")



#-----------------------------------------------------------------------------------------------
### cancer progression trajectory from Seurat object
#-----------------------------------------------------------------------------------------------


library(monocle)


# isolation of specific cell types
```

```
Fol<-subset(data.combined5, cells=names(data.combined5@active.ident[data.combined5@active.ident %in% c("Follicular")]))
Fol<-subset(Fol, cells=names(Fol@active.ident[Fol$SubType%in% c("NOM","PTC","ATC")]))

ATC<-subset(data.combined5, cells=names(data.combined5@active.ident[data.combined5@active.ident %in% c("ATC")]))
ATC<-subset(ATC, cells=names(ATC@active.ident[ATC$SubType %in% c("ATC","PTC")]))

PTC<-subset(data.combined5, cells=names(data.combined5@active.ident[data.combined5$orig.ident %in% c("PTC_XHY1_190702",
"PTC_XHY2_190702")]))
PTC<-subset(PTC, cells=names(PTC@active.ident[PTC@active.ident %in% c("PTC")]))

aPTC<-subset(data.combined5, cells=names(data.combined5@active.ident[data.combined5@active.ident %in% c("PTC")]))
aPTC<-subset(aPTC, cells=names(aPTC@active.ident[aPTC$SubType %in% c("ATC")]))

PTC2<-subset(data.combined5, cells=names(data.combined5@active.ident[data.combined5@active.ident %in% c("PTC2")]))
PTC2<-subset(PTC2, cells=names(PTC2@active.ident[PTC2$SubType %in% c("PTC")]))

a<-subset(Fol, cells=names(Fol@active.ident[Fol$SubType %in% c("NOM")]))
b<-subset(Fol, cells=names(Fol@active.ident[Fol$SubType %in% c("PTC")]))
c<-subset(Fol, cells=names(Fol@active.ident[Fol$SubType %in% c("ATC")]))

e<-subset(ATC, cells=names(ATC@active.ident[ATC$SubType %in% c("PTC")]))
f<-subset(ATC, cells=names(ATC@active.ident[ATC$SubType %in% c("ATC")]))

Idents(object =Fol, cells =colnames(a))="NOM_follicular"
Idents(object =Fol, cells =colnames(b))="PTC_follicular"
Idents(object =Fol, cells =colnames(c))="ATC_follicular"

Idents(object =ATC, cells =colnames(e))="pATC"
Idents(object =ATC, cells =colnames(f))="aATC"

Idents(object =PTC, cells =colnames(PTC))="pPTC"

Idents(object =aPTC, cells =colnames(aPTC))="aPTC"

table(Fol@active.ident)
table(ATC@active.ident)

thyroid <- merge(Fol, y = c(ATC,PTC,PTC2,aPTC), project = "thyroid")
```

 # monocle2 workflow

```
table(thyroid@active.ident)

thyroid$nCount_RNA=NULL

thyroid$nFeature_RNA=NULL

thyroid$percent.mt=NULL

thyroid$orig.ident<-thyroid@active.ident

thyroid@meta.data=thyroid@meta.data[,1:3]

x=GetAssayData(object = thyroid, slot = "counts")[rownames(GetAssayData(object = thyroid, slot = "counts")) %in% rownames(GetAssayData(object =
thyroid)), colnames(GetAssayData(object = thyroid, slot = "counts")) %in% colnames(GetAssayData(object = thyroid))]

target=x[,colnames(x) %in% rownames(thyroid@meta.data)]

target<-as.matrix(target)

target=as.matrix(t(target))

name=merge(thyroid@meta.data, target, by="row.names")

rownames(name)=name$Row.names

name=name[,-1]

name[1:5,1:4]
```

```
name[1:5,1:4]

thyroid@meta.data=name[,1:4]

name=name[,-1:-3]

target=t(name)

target[1:5,1:4]

g=as.data.frame(rownames(x))

rownames(g)=rownames(x)

colnames(g)=c("gene_short_name")



pd <- new("AnnotatedDataFrame", data = thyroid@meta.data)

fd <- new("AnnotatedDataFrame", data = g)

HSMM <- newCellDataSet(as.matrix(target), phenoData = pd, featureData = fd, expressionFamily=negbinomial.size())

HSMM <- estimateSizeFactors(HSMM)

HSMM <- estimateDispersions(HSMM)

HSMM <- detectGenes(HSMM, min_expr = 0.1)

print(head(fData(HSMM)))

expressed_genes <- row.names(subset(fData(HSMM), num_cells_expressed >= 10))

disp_table <- dispersionTable(HSMM)

ordering_genes <- subset(disp_table, mean_expression >= 0.05 & dispersion_empirical >= 2 * dispersion_fit)$gene_id

HSMM <- setOrderingFilter(HSMM, ordering_genes)

plot_ordering_genes(HSMM)

HSMM <- reduceDimension(HSMM, max_components=2)

HSMM <- orderCells(HSMM, reverse=FALSE)



coordinate<-t(HSMM@reducedDimS)

head(coordinate)

colnames(coordinate)<-c("x","y")

coordinate<-as.data.frame(coordinate)

coordinate<-cbind(-coordinate$y,coordinate$x)

coordinate<-t(coordinate)

HSMM@reducedDimS<-coordinate

colnames(HSMM@reducedDimS)<-colnames(HSMM)



plot_cell_trajectory(HSMM, show_tree = FALSE ,show_backbone = FALSE, color_by="orig.ident", show_branch_points = FALSE, cell_size=1)

plot_cell_trajectory(HSMM, show_tree = FALSE ,show_backbone = FALSE, color_by="orig.ident", show_branch_points = FALSE, cell_size=1) +
scale  color  manual(breaks = c("aPTC", "pATC","ATC" ,"PTC", "PTC2", "NOM  follicular", "PTC  follicular", "ATC  follicular"), values=c("black",
```

"darkorchid","darkseagreen2", "dodgerblue3", "indianred3","yellow2","turquoise3","RED1")) + theme(legend.position = "right")

#----------
gene expression on the trajectory
#----------

pseudo=t(HSMM@reducedDimS)

colnames(pseudo)=c("component1","component2")

expression=target["COL1A1",]

expression[expression>5]=5

ggplot() + geom_point(data=as.data.frame(pseudo), aes(component1, component2, color=expression), size=1.5) + scale_colour_gradient(low = "grey",high = "red")

#----------
expression change along the pseudotime
#----------

HSMM_subset <- HSMM2[,colnames(HSMM2)%in% rownames(subset(pData(HSMM2), State %in% c("1","2")))]

HSMM_subset@reducedDimS<-HSMM_subset@reducedDimS[,colnames(HSMM_subset@reducedDimS)%in%colnames(HSMM_subset)]

HSMM_expressed_genes <- row.names(subset(fData(HSMM_subset), num_cells_expressed >= 10))

HSMM_expressed_genes <- row.names(subset(fData(HSMM), num_cells_expressed >= 10))

HSMM_filtered <- HSMM[HSMM_expressed_genes,]

HSMM_filtered <- HSMM_subset[HSMM_expressed_genes,]

diff_test_res <- differentialGeneTest(HSMM_filtered, fullModelFormulaStr="~sm.ns(Pseudotime)")

diff=diff_test_res[,c("gene_short_name", "pval", "qval","use_for_ordering")]

diff=diff[diff$qval < 0.01,]

diff=diff[order(diff$qval),]

genelist <- row.names(diff_test_res[order(diff_test_res$qval)[1:800],])

genelist<-as.matrix(genelist)

plot_pseudotime_heatmap(HSMM[genelist,],num_clusters = 2, cores =1, show_rownames = T)

#----------
extraction gene list
#----------

pseudotime_list<-plot_pseudotime_heatmap(HSMM[genelist,],num_clusters = 2, cores = 1,show_rownames = T, return_heatmap = T)

pseudotime_list <- as.data.frame(cutree(pseudotime_list $tree_row, k=2))

colnames(pseudotime_list) <- "Cluster"

pseudotime_list $Gene <- rownames(pseudotime_list)

head(pseudotime_list)

```
write.csv(pseudotime_list,file ="coordinate.csv", sep = "," )
```

How to cite:(Readers should cite both the Bio-protocol preprint and the original research article where this protocol was used)

1. Luo, H. , Kim, G. , Wei, T. , Park, J. and Xu, H. (2021). Single-cell bioinformatics analysis for thyroid cancer study. Bio-protocol Preprint. bio-protocol.org/prep1332.
2. Luo, H., Xia, X., Kim, G. D., Liu, Y., Xue, Z., Zhang, L., Shu, Y., Yang, T., Chen, Y., Zhang, S., Chen, H., Zhang, W., Li, R., Tang, H., Dong, B., Fu, X., Cheng, W., Zhang, W., Yang, L., Peng, Y., Dai, L., Hu, H., Jiang, Y., Gong, C., Hu, Y., Zhu, J., Li, Z., Caulin, C., Wei, T., Park, J. and Xu, H.(2021). Characterizing dedifferentiation of thyroid cancer by integrated analysis . Science Advances 7(31). DOI: 10.1126/sciadv.abf3657